

StyleVideoGAN: A Temporal Generative Model using a Pretrained StyleGAN – SUPPLEMENTAL MATERIAL –

Gereon Fox

gfox@mpi-inf.mpg.de

Ayush Tewari

atewari@mpi-inf.mpg.de

Mohamed Elgharib

elgharib@mpi-inf.mpg.de

Christian Theobalt

theobalt@mpi-inf.mpg.de

Max Planck Institute for Informatics

Saarland Informatics Campus

Saarbrücken, Germany

1 Quantitative results for additional identities

In Tables 1 and 2 we give more quantitative results, on the additional identities depicted in Fig. 1. Our method outperforms the state of the art for these subjects as well.

Model	Reference	FID (\downarrow)		FVD (\downarrow)	
		Short	Long	Short	Long
Ours	Original	62.9 ± 0.2	65.1 ± 0.8	846.7 ± 19.1	944.1 ± 51.7
	\mathcal{W}^+	0.6 ± 0.0	2.0 ± 0.2	39.2 ± 19.2	51.8 ± 18.1
Ours $\setminus \mathcal{L}_{\text{GAP}}$	Original	62.9 ± 0.1	64.8 ± 3.0	848.8 ± 8.0	926.2 ± 58.5
	\mathcal{W}^+	0.7 ± 0.0	4.3 ± 2.8	57.0 ± 93.1	110.7 ± 74.9
Tulyakov [10]	Original	76.5	77.8	1318.7	1338.7
Saito [9]	Original	41.5	51.6	640.0	935.2
Munoz [3]	Original	46.4	-	578.3	-

Table 1: FID and FVD scores for subject # 2. All metrics were computed as described in the main paper.

2 Experimental Details

2.1 Training details

We trained all methods with their default hyperparameters, except for Saito et al, where we had to adjust the batch size to 2 and set `clstm_channels = 512`. The authors of

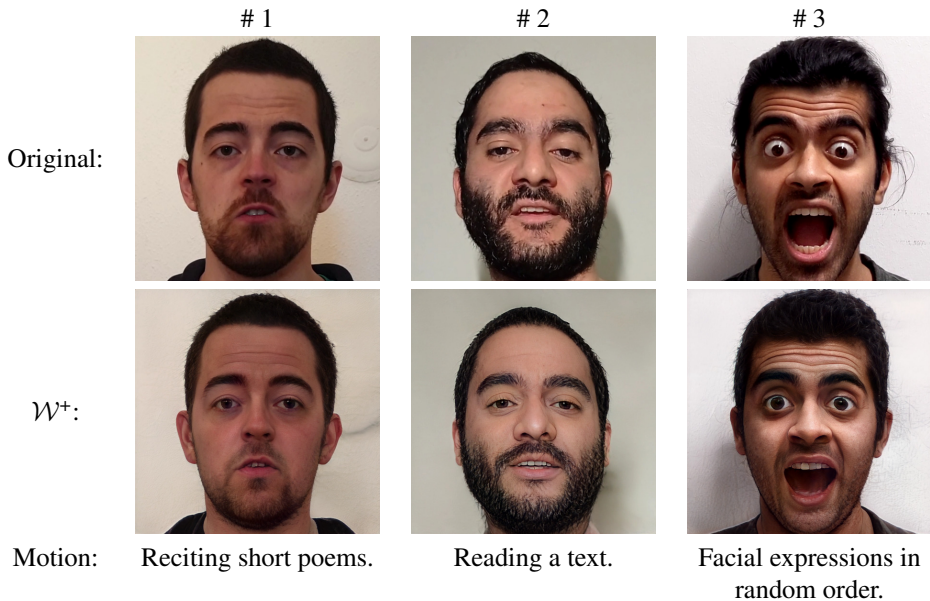


Figure 1: Screenshots from the training sequences we used for quantitative evaluation. The numbers in this supplemental document have been computed for subjects # 2 and # 3. The ones reported in the main paper are for # 1.

Tian *et al.* [15] kindly trained their technique on the training data we sent them. We trained all methods with at least the computational resources that our method uses, but usually gave them much more training time. Table 3 gives a comprehensive overview.

Each training video contains 1 single actor/object. We computed quantitative evaluations for 3 actors, reported in the main paper and this supplemental. For the proof of concept on hands, all training data was recorded from one actor. In the case of cars, we show qualitative results of 3 different cars. We use a batch size of 128, learning rate of 0.005, and exponential averaging of the weights with a momentum of 0.997 in all experiments. All temporal generative networks are trained for 350 epochs.

2.2 Evaluation details

FID scores have been computed by sampling 8000 frames from both the training set (as preprocessed for the particular method) and the set of generated videos. We used the FID implementation in <https://github.com/mseitzer/pytorch-fid>.

FVD scores have been computed by sampling 2048 video slices from both the training set (as preprocessed for the particular method) and the set of generated videos. For each method, and regardless of the length of the samples we generated (“short” versus “long”), the videos we sampled were always 25 frames long for our method and 16 frames long for the previous methods. We used the FVD implementation in

https://github.com/google-research/google-research/tree/master/frechet_video_distance

Model	Reference	FID (\downarrow)		FVD (\downarrow)	
		Short	Long	Short	Long
Ours	Original	52.7 \pm 0.2	53.7 \pm 0.5	589.2 \pm 9.7	625.3 \pm 3.1
	\mathcal{W}^+	3.7 \pm 0.2	4.8 \pm 0.4	61.4 \pm 4.2	98.6 \pm 11.3
Ours $\setminus \mathcal{L}_{\text{GAP}}$	Original	52.3 \pm 0.5	55.2 \pm 2.1	590.9 \pm 15.2	679.0 \pm 28.9
	\mathcal{W}^+	3.4 \pm 0.1	8.2 \pm 2.0	54.0 \pm 3.6	133.5 \pm 28.5
Tulyakov [10]	Original	123.0	141.1	1163.3	1500.3
Saito [15]	Original	82.1	270.3	823.9	2090.8
Munoz [16]	Original	83.3	-	1037.0	-

Table 2: FID and FVD scores for subject # 3. This subject was not talking, but instead performing some simple face motions in a random order (like smiling or acting surprised). The training video is only 1 minute and 20 seconds in length.

	GPU Used	GPU Memory	Training time (max)
Ours	Quadro RTX 8000	48 GB (8GB used)	approx. 6 hours
Tulyakov <i>et al.</i> [10]	GeForce RTX 2080	12 GB	approx. 6 hours
Munoz <i>et al.</i> [16]	Quadro RTX 8000	48 GB	approx. 2 days
Saito <i>et al.</i> [15]	GeForce RTX 2080	12 GB	approx. 15 hours

Table 3: Training details for the various methods. The authors of Tian *et al.* [14] trained their method for us.

ACD scores have been computed always on 128 “long” samples drawn from the trained models. For our method these long samples were 400 frames long. For Tian *et al.* they were 128 frames long. Having a good ACD becomes harder as sequences grow longer.

3 Limitations

Even though we are able to further to the state of the art in video generation, in particular with respect to the amounts of computational resources and training data necessary to generate a large amount of diverse videos, our method has several limitations: For one, the quality of our generated videos strictly depends on the quality of the underlying StyleGAN model and its corresponding pSp inverter. For example, in the case of faces we have observed that nontrivial video backgrounds tend to not be represented in a temporally stable way. The importance of temporally stable embedding is also underlined by an experiment we made with a BigGAN model [17] instead of a StyleGAN model: We used a SOTA optimization-based method [18] to embed several short videos (to the best of our knowledge there are no encoder-based inversion methods, see [19]). The embeddings contain strong temporal noise, making training our method pointless (see supplemental video). For the sources of the videos, see Table 8

Another limitation is the fact that while our approach does not contain any inherently face-specific components and even though we are showing a proof-of-concept for animating hands and cars, it is still unclear whether all the advantageous properties of StyleGAN’s \mathcal{W}^+ space can be made use of in any arbitrary domain, e.g. if our offset trick will work there.

4 Detailed architecture

For the sake of completeness, Tables 4 to 7 give detailed specifications for the architecture components that we outlined in Figure 2 of the main paper.

Input	Module	Outputs (Dimensionality)
i	4 layers (Fully Connected + LeakyReLU)	m (3×32)
m	BatchNorm	$(h_{0,0}, h_{0,1}, h_{0,2})$ (3×32)

Table 4: The “hallucinator” H is responsible for producing some initial contents for the GRU memory. For each one of the four stacked GRU cells it produces a vector of length 32.

Input	Module	Outputs (Dimensionality)
$s_0, (h_{0,0}, \dots, h_{0,3})$	GRU (4 stacked cells)	$(h_{1,0}, \dots, h_{1,3}), l_1$ ($3 \times 32, 32$)
$s_1, (h_{1,0}, \dots, h_{1,3})$	GRU (4 stacked cells)	$(h_{2,0}, \dots, h_{2,3}), l_2$ ($3 \times 32, 32$)
$s_2, (h_{2,0}, \dots, h_{2,3})$	GRU (4 stacked cells)	$(h_{3,0}, \dots, h_{3,3}), l_3$ ($3 \times 32, 32$)
\dots	\dots	\dots

Table 5: The feature generator P consists of four stacked GRU cells. Its hidden state is initialized with the output of H (Table 4) and it translates a sequence of random vectors s_k into intermediate latent codes l_{k+1} for $0 \leq k < t - 1$ in a recurrent fashion.

Input	Module	Outputs (Dimensionality)
l_k	BatchNorm + Affine transform + Pixel-Norm	l'_k (512)
l'_k	4 layers (FullyConnected + LeakyReLU)	v'_k (512)
v'_k	BatchNorm + Affine transform	v_k (512)
v_k	18 parallel layers (FullyConnected + LeakyReLU + BatchNorm)	w_k (18×512)

Table 6: The latent mapper T is an MLP that transforms the outputs l_k of the feature generator P into StyleGAN latent codes $w_k \in \mathcal{W}^+$: After a 4-layer MLP that widens the dimensionality from 32 to 512, we employ 18 independent fully connected layers (with LeakyReLU activation), in a way similar to how StyleGAN broadcasts its \mathcal{W} vectors to its 18 Style layers.

Input	Module	Outputs (Dimensionality)
w_k	FullyConnected + LeakyReLU (2 layers)	e'_k (512)
e'_k	FullyConnected + LeakyReLU (4 layers)	e_k (32)
e_0, \dots, e_{t-1}	1D-version of the DCGAN critic [4], with 32 input channels	Critic Output (1)

Table 7: The latent critic takes a sequence w_0, \dots, w_{t-1} of StyleGAN latent codes as input, and produces a scalar output.

BigGAN category	YouTube key
indigo bird (014)	DZOiCmxSU2k
green mamba (064)	hG4Wvp0U18A
bison (347)	L4eOhuLDfeU
gazelle (353)	jMIiB9DnRXg

Table 8: The sequences we embedded into the BigGAN latent space were excerpts of YouTube videos.

References

- [1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.
- [2] Minyoung Huh, Jun-Yan Zhu Richard Zhang, Sylvain Paris, and Aaron Hertzmann. Transforming and projecting images to class-conditional generative networks. In *ECCV*, 2020.
- [3] Andres Munoz, Mohammadreza Zolfaghari, Max Argus, and Thomas Brox. Temporal shift GAN for large scale video generation. In *WACV*, 2021.
- [4] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.
- [5] Masaki Saito, Shunta Saito, Masanori Koyama, and Sosuke Kobayashi. Train sparsely, generate densely: Memory-efficient unsupervised training of high-resolution temporal GAN. *IJCV*, 128(10-11), 2020.
- [6] Yu Tian, Jian Ren, Menglei Chai, Kyle Olszewski, Xi Peng, Dimitris N. Metaxas, and Sergey Tulyakov. A good image generator is what you need for high-resolution video synthesis. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=6puCSjH3hwa>.
- [7] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing motion and content for video generation. In *CVPR*, 2018.
- [8] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. GAN inversion: A survey. *CoRR*, abs/2101.05278, 2021. URL <https://arxiv.org/abs/2101.05278>.